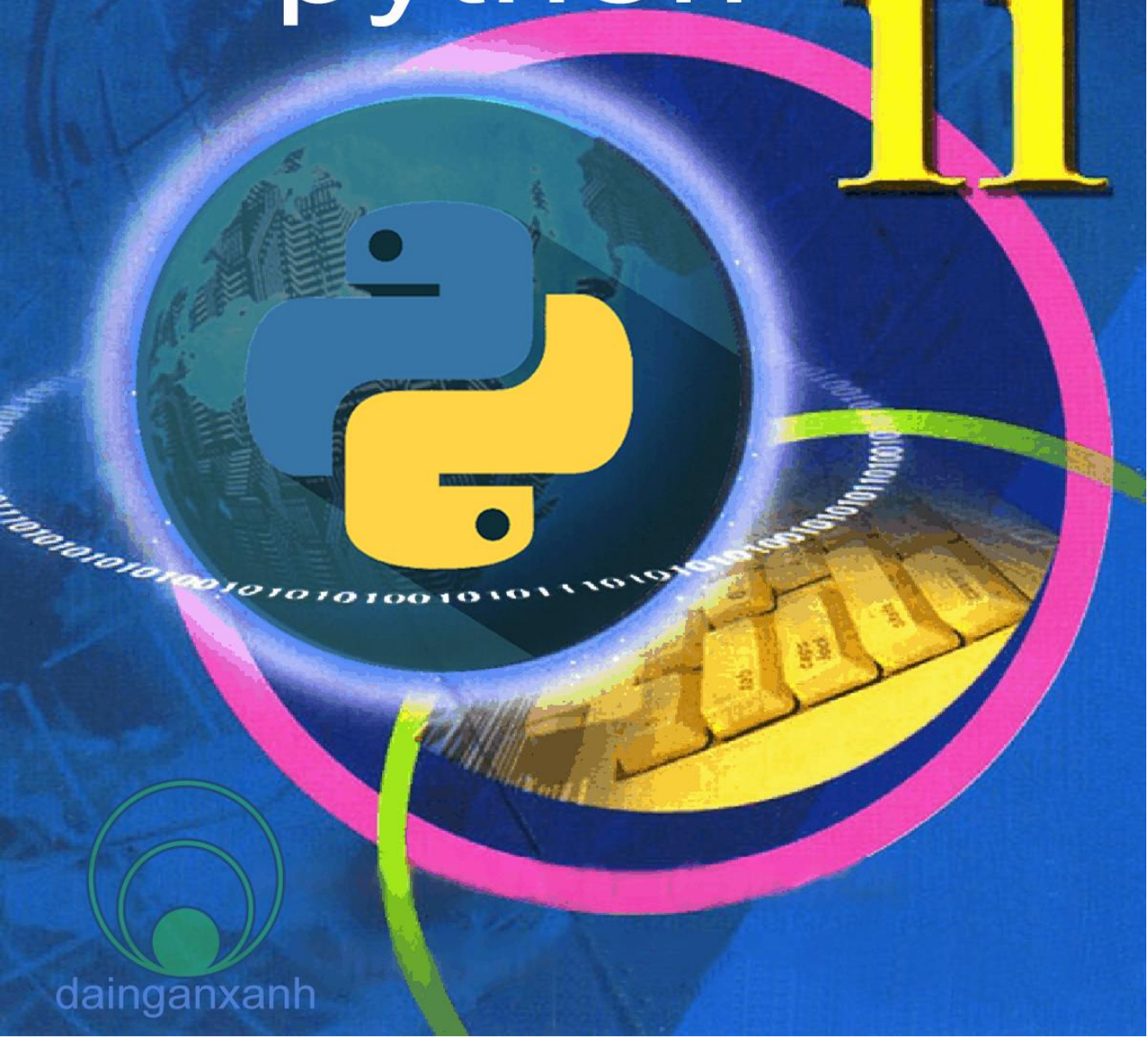
BIÊN SOẠN THEO SÁCH GIÁO KHOA CỦA NXB GIÁO DỤC

TIN HỌC

python

11



dainganxanh

ThS. NGUYỄN VĂN NGHIÊM

SÁCH DÀNH CHO GIÁO VIÊN

****

Hè 2021

Python là gì ?

Python là một ngôn ngữ lập trình bậc cao do Guido van Rossum tạo ra và lần đầu ra mắt vào năm 1991. Python vừa hướng thủ tục (procedural-oriented), vừa hướng đối tượng (object-oriented) đồng thời có thể nhúng vào ứng dụng như một giao tiếp kịch bản (scripting interface).

Thế mạnh của Python là rất gần gũi với ngôn ngữ tự nhiên (tiếng Anh), cấu trúc rõ ràng, dễ đọc, dễ học. Python hiện nay là ngôn ngữ lập trình phổ biến rộng rãi ở châu Âu, châu Mỹ và được coi như ***ngôn ngữ lập trình trường học***.

Python được dùng để phát triển các ứng dụng web, game, khoa học dữ liệu (tính toán, phân tích, khai thác dữ liệu), máy học và trí tuệ nhân tạo, …

Tài liệu dùng kèm Sách giáo khoa Tin học 11 của Bộ Giáo dục và Đào tạo (NXB. Giáo Dục - Tái bản lần thứ 4, năm 2009).

Tài liệu được trình bày theo cấu trúc Sách giáo khoa Tin học 11. Các ví dụ và bài tập, bài thực hành trong sách giáo khoa được trình bày lại bằng ngôn ngữ lập trình Python một cách chi tiết, đầy đủ.

Bổ sung một số kiến thức, kỹ thuật lập trình cần thiết để sử dụng ngôn ngữ lập trình Python trong dạy và học chương trình Tin học 11.

MỤC LỤC

LỜI NÓI ĐẦU 1
1. Hướng dẫn đọc 1
2. Cài Python 1
3. Cài chương trình soạn thảo 1
Chương 1. MỘT SỐ KHÁI NIỆM 3
§1. KHÁI NIỆM VỀ LẬP TRÌNH VÀ NGÔN NGỮ LẬP TRÌNH 3
1. Lưu ý 3
§2. CÁC THÀNH PHẦN CỦA NGÔN NGỮ LẬP TRÌNH 4
1. Các thành phần cơ bản 4
2. Một số khái niệm 4
Chương 2. CHƯƠNG TRÌNH ĐƠN GIẢN 6
§3. CẤU TRÚC CHƯƠNG TRÌNH 6
1. Câu trúc chung 6
2. Các thành phần của chương trình 6
3. Ví dụ chương tình đơn giản 7
§4. MỘT SỐ KIỂU DỮ LIỆU CHUẨN 8
1. Kiểu nguyên 8
2. Kiểu thực 8
3. Kiểu kí tự 8
4. Kiểu logic 9
§5. KHAI BÁO BIẾN 10
§6. PHÉP TOÁN, BIỂU THỨC, CÂU LỆNH GÁN 11
1. Phép toán 11
2. Biểu thức số học 11
3. Hàm số học chuẩn 12
4. Biểu thức quan hệ 12
5. Biểu thức logic 12

6. Câu lệnh gán 12
§7. CÁC THỦ TỤC CHUẨN VÀO RA ĐƠN GIẢN 14
1. Nhập dữ liệu vào từ bàn phím 14
2. Đưa dữ liệu ra màn hình 14
§8. SOẠN THẢO, DỊCH, THỰC HIỆN VÀ HIỆU CHỈNH CHƯƠNG TRÌNH 16
BÀI TẬP VÀ THỰC HÀNH 1 18
1. Mục đích, yêu cầu 18
2. Nội dung 18
Chương 3. CẤU TRÚC RẼ NHÁNH VÀ LẶP 20
§9. CẤU TRÚC RẼ NHANH 20
1. Rẽ nhánh 20
2. Câu lệnh if 20
3. Câu lệnh ghép 22
4. Một số ví dụ 22
§10. CẤU TRÚC LẶP 24
1. Lặp 24
2. Lặp với số lần biết trước 24
3. Lặp với số lần chưa biết trước 25
BÀI TẬP VÀ THỰC HÀNH 2 27
Chương 4. KIỂU DỮ LIỆU CÓ CẤU TRÚC 31
§11. KIỂU MẢNG 31
1. Kiểu mảng một chiều 32
2. Kiểu mảng hai chiều 38
3. Các thao tác xử lý list 40
BÀI TẬP VÀ THỰC HÀNH 3 44
BÀI TẬP VÀ THỰC HÀNH 4 46
§12. KIỂU XÂU 48
1. Khai báo 48

2. Các thao tác xử lý xâu 48
3. Một số ví dụ 53
BÀI TẬP VÀ THỰC HÀNH 5 56
§13. KIỂU BẢN GHI 58
1. Khai báo 58
2. Gán giá trị 58
3. Các thao tác cơ bản với kiểu dict 60
CÂU HỎI VÀ BÀI TẬP 62
Chương 5. TỆP VÀ THAO TÁC VỚI TỆP 67
§14. KIỂU DỮ LIỆU TỆP 67
1. Vai trò của kiểu tệp 67
2. Phân loại tệp và thao tác với tệp 67
§15. THAO TÁC VỚI TỆP 68
1. Khai báo 68
2. Thao tác với tệp 68
§16. VÍ DỤ LÀM VIỆC VỚI TỆP 71
Chương 6. CHƯƠNG TRÌNH CON 73
§17. CHƯƠNG TRÌNH CON VÀ PHÂN LOẠI 73
1. Khái niệm chương trình con 73
2. Phân loại và cấu trúc của chương trình con 74
§18. VÍ DỤ CÁCH VIẾT VÀ SỬ DỤNG CHƯƠNG TRÌNH CON. 79
BÀI TẬP VÀ THỰC HÀNH 6 82
1. Mục đích, yêu cầu 82
2. Nội dung 82
BÀI TẬP VÀ THỰC HÀNH 7 83
1. Mục đích, yêu cầu 83
2. Nội dung 83
§19. THƯ VIỆN CHƯƠNG TRÌNH CON CHUẨN 87

1. Module 87
2. Package 88
3. Thư viện đồ họa Turtle..... 88
BÀI TẬP VÀ THỰC HÀNH 7.....91
1. Mục đích, yêu cầu 91
2. Nội dung..... 91
Tìm hiểu thêm về Python 95

LỜI NÓI ĐẦU

1. Hướng dẫn đọc

Sách này dùng kèm với sách giáo khoa Tin học 11 của Bộ Giáo dục và Đào tạo, Nhà xuất bản Giáo dục Việt Năm tái bản lần thứ 4, năm 2009 (SGK).

Những phần chỉ có tiêu đề, không có nội dung nghĩa là sử dụng nguyên văn nội dung được trình bày trong SGK (các nội dung được trình bày trong SGK là phù hợp với Python).

Nội dung sách này bám sát theo đúng cấu trúc của SGK. Vì vậy sách chỉ trình bày và giải quyết các vấn đề được SGK nêu - đó chỉ là một phần rất cơ bản của lập trình và ngôn ngữ lập trình. Để có thể lập trình và làm chủ ngôn ngữ lập trình Python thì chúng ta cần nghiên cứu thêm các tài liệu khác (có giới thiệu ở cuối sách).

2. Cài Python

Tải về từ https://www.python.org/downloads/ và tiến hành cài đặt (chọn phiên bản 3.8 trở lên).

Sau khi hoàn tất cài đặt có thể kiểm tra:

- Nhấn phím Windows gõ cmd → Enter
- Gõ: python --version → Enter

Lúc này sẽ hiển thị phiên bản Python đã cài đặt trên máy tính.

3. Cài chương trình soạn thảo

Để lập trình theo một ngôn ngữ nào đó ta đều cần có chương trình cho phép gõ các câu lệnh và ra lệnh thực thi các câu lệnh đó. Trong các trường học, để lập trình với Pascal ta thường sử dụng FreePascal, với C ta thường dùng CodeBlock, … Với Python, ta có nhiều lựa chọn. Dưới đây là một số gợi ý:

1) Python's IDLE

Python's IDLE là từ viết tắt của cụm từ "Python's Integrated Development and Learning Environment", là một công cụ, môi trường tích hợp sẵn phục phát triển và học tập Python. IDLE được tích hợp sẵn trong bộ cài đặt Python.

2) Notepad++

Tải về tại đây: https://notepad-plus-plus.org/downloads/

Đặc điểm: Đơn giản, dễ sử dụng.

Nhược điểm: Phải cài thêm plugin để debug.

3) Thonny

Tải về tại đây: https://thonny.org/

Thonny có giao diện đơn giản, cấu hình nhẹ (trên cùng một máy khởi động nhanh hơn nhiều so với Pycham hay Spyder). Hỗ trợ debug trực quan giúp ta dễ theo dõi và hình dung quá trình thực thi chương trình. Sử dụng thư viện / module chuẩn của Python phát hành (không bổ sung hay import sẵn module).

4) PyCharm Educational Edition

Tải về tại đây: https://www.jetbrains.com/pycharm-edu/

PyCharm là môi trường phát triển tích hợp đa nền tảng (IDE) được phát triển bởi Jet Brains và được thiết kế đặc biệt cho Python. Tuy nhiên PyCharm khởi động khá nặng nề và yêu cầu làm việc với project.

Như vậy, tùy nhu cầu sử dụng và kỹ năng lập trình mà chúng ta lựa chọn trình soạn thảo cho phù hợp. Đối với người mới bắt đầu học Python thì nên dùng Thonny để thực hành.

Chương 1

MỘT SỐ KHÁI NIỆM VỀ LẬP TRÌNH VÀ NGÔN NGỮ LẬP TRÌNH

§1. KHÁI NIỆM VỀ LẬP TRÌNH VÀ NGÔN NGỮ LẬP TRÌNH

1. Lưu ý

Python là một ngôn ngữ lập trình phiên dịch (Interpreter Language), tức là không cần build thành file thực thi mà chạy trực tiếp. *(Pascal và C++ phải build trước khi thực thi).*

Các nội dung còn lại được trình bày của phần này trong SGK Tin học 11 là phù hợp với Python.

§2. CÁC THÀNH PHẦN CỦA NGÔN NGỮ LẬP TRÌNH

1. Các thành phần cơ bản

Nội dung trình bày của phần này trong SGK Tin học 11 là phù hợp với Python.

Lưu ý thêm: Python sử dụng mặc định mảng mã Unicode thay vì ASCII như Pascal. Như vậy, khái niệm "bảng chữ cái" trong Python là các ký tự trong bảng mã Unicode. Tên biến và các đối tượng trong chương trình Python có để đặt bằng tiếng Việt có dấu.

2. Một số khái niệm

a) Tên

Trong Python, Tên các đối tượng được đặt bằng các ký tự thường (a-z), ký tự in hoa (A-Z), chữ số (0-9) và dấu gạch dưới _ .

Tên đối tượng không bắt đầu bằng chữ số, không dùng các ký tự đặc biệt như !, @, #, … và được phân biệt chữ hoa, chữ thường.

Tên trong Python không giới hạn độ dài. Tuy nhiên nên đặt tên có tính gợi nhớ về đối tượng. Ví dụ đặt tên biến để lưu giá trị đếm số lần thực thi thì nên đặt là "dem" hay "count",…

Tên dành riêng

Tên dành riêng được hiểu là Từ khóa (keyword) trong Python.

Keyword được định nghĩa sẵn để sử dụng. Chúng ta không thể dùng keyword để đặt tên biến, tên hàm hoặc bất kỳ đối tượng nào trong chương trình.

Tất cả các keyword trong Python đều được viết thường, trừ 03 keyword: True, False, None.

Ví dụ một số keyword: True, False, await, else, import, pass, break, except, in, and, or, ….

Thực tế thì không cần nhớ keyword vì khi gõ trình soạn thảo sẽ có gợi ý các keyword (ta tránh đặt tên đối tượng trùng với các keyword được gợi ý) và nếu đặt trùng tên keyword thì khi chạy chương trình sẽ báo lỗi.

Tên chuẩn

Tên chuẩn là những tên đã được định nghĩa (tên module) thuộc thư viện chuẩn của Python.

Ví dụ: math (module các hàm toán học thông dụng như sin, cos, tan, sqr, sqrt,…).

b) Hằng và biến

Trong phần này ta nên nói về biến trước khi nói về hằng.

Biến

Biến là đại lượng (đối tượng) được đặt tên, dùng để lưu trữ giá trị và giá trị có thể được thay đổi trong quá trình thực hiện chương trình.

Biến trong Python không cần khai báo trước, không nhất thiết phải khai báo kiểu dữ liệu. Khi đặt tên và gán giá trị Python tự động nhận dạng và tùy biến theo kiểu dữ liệu được gán.

Hằng

Hằng là một loại biến đặc biệt, giá trị của hằng là không đổi trong suốt chương trình sau lần gán giá trị đầu tiên. Tên hằng được viết hoàn toàn bằng CHỮ HOA và dấu gạch dưới (nếu cần).

Trong thực hành, hằng thường được khai báo trong một file riêng biệt (không phải file chương trình) như một module và được import vào chương trình để sử dụng.

c) Chú thích

Chú thích trong Python có thể sử dụng các cách sau:

```
# dùng dấu thăng đầu dòng khi chú thích trên một dòng.
''' dùng ba dấu nháy đơn hoặc nháy kép
Khi chú thích trên nhiều dòng
'''
```

Chương 11

CHƯƠNG TRÌNH ĐƠN GIẢN

§3. CẤU TRÚC CHƯƠNG TRÌNH

1. Câu trúc chung

Chương trình Python không phân chia 2 phần như C, Pascal. Một chương trình đơn giản có thể chỉ có một dòng lệnh. Tuy nhiên, trong một số trường hợp ta vẫn thường khai báo sử dụng chương trình con (module) ở đầu chương trình.

Ta cũng có thể thay phần khai báo bằng một ghi chú về chương trình.

2. Các thành phần của chương trình

2.1 Phần khai báo

Không phải khi nào cũng cần có phần khai báo. Chỉ khai báo khi chương trình có sử dụng đến thư viện chương trình con (module) nào đó.

Ví dụ:

```
# khai báo dùng thư viện các hàm toán học
import math
```

Khai báo biến, hằng không nhất thiết phải khai báo ở đầu chương trình mà có thể khai báo bất cứ vị trí nào trong chương trình trước khi dùng đến:

```
a = 0
TONG = 100
```

2.2 Phần thân chương trình

Phần thân chương trình Python là các câu lệnh thực thi.

Tóm lại, Python không phân biệt phần khai báo và phần thân như C hay Pascal. Một chương tình đơn giản của Python có thể chỉ có một dòng lệnh như Ví dụ 1 dưới đây.

3. Ví dụ chương tình đơn giản

Ví dụ 1.

```
print('Xin chào các bạn! ')
```

Ví dụ 2.

```
print('xin chào các bạn! ')
print('Mời các bạn làm quen với Python')
```

Xin lưu ý, code trong Python mặc định xâu ký tự ở chuẩn unicode nên có thể gõ tiếng Việt có dấu thoải mái.

§4. MỘT SỐ KIỂU DỮ LIỆU CHUẨN

Trong Python có 6 kiểu dữ liệu chuẩn gồm: Numbers, String, List, Tuple, Set, Dictionary.

Trong bài này chúng ta tìm hiểu các kiểu dữ liệu được trình bày trong Bài 4, SGK Tin học 11. Những kiểu dữ liệu khác sẽ được trình bày trong bài khác hoặc phần đọc thêm cuối sách.

1. Kiểu nguyên

Kiểu nguyên (int) trong Python không giới hạn số ký tự mà chỉ phụ thuộc vào bộ nhớ máy tính.

Khi gán một giá trị là số nguyên cho một biến thì biến đó tự động được gán kiểu số nguyên.

```
# gán cho biến a một số nguyên có giá trị là 5.5 ➔ a sẽ nhận giá trị là 5.
a = int(5.5)
b = 120
```

2. Kiểu thực

Kiểu thực (float) trong Python có giới hạn tối đa 15 chữ số phần thập phân.

Khi gán một giá trị là số thực cho một biến thì biến đó tự động được gán kiểu số thực.

```
# gán cho a số thực có giá trị là 5 ➔ a sẽ nhận giá trị là 5.0
a = float(5)
b = 3.14
```

3. Kiểu kí tự

Python không có kiểu char như Pascal. Một ký tự (kiểu char của pascal) được coi như một xâu có độ dài bằng 1 trong Python. Kiểu xâu (str) hay còn gọi là kiểu chuỗi không giới hạn độ dài.

Tuy nhiên, Python cung cấp các hàm chr() và ord() để lấy vị trí của ký tự trong bảng mã Unicode và ngược lại. Chương trình dưới đây cho thấy mã của chữ Â trong bảng mã unicode là 194.

```
print(ord('Â'))
print(chr(194))
```

Kết quả:

```
194
Â
```

4. Kiểu logic

Kiểu logic (bool) trong Python có giá trị True hoặc False

Ngoài ra, Python còn có các kiểu dữ liệu: complex; list, tuple, range; dict; set, frozenset; bytes, bytearray, memoryview. Trong phần sau của chương trình chúng ta sẽ tìm hiểu thêm.

§5. KHAI BÁO BIẾN

Trong Python một biến không cần khai báo kiểu dữ liệu. Khi ta gán giá trị thì tự động Python sẽ tùy biến kiểu dữ liệu của biến cho phù hợp với dữ liệu được gán vào.

Ví dụ trong cùng một chương trình khai báo như sau biến a sẽ tự động chuyển đổi kiểu để lưu giá trị được gán:

```
a = 'Học Python'   # biến a có kiểu xâu
a = 5              # biến a đổi sang kiểu nguyên
a = 5.5            # biến a đổi sang kiểu thực
a = True           # biến a đổi sang kiểu logic
```

Ta có thể ép kiểu cho biến bằng cách khai báo kiểu cho giá trị gán cho biến.

Ví dụ:

```
a = str(5)         # a là một biến kiểu xâu
b = int(5.5)       # b là một biến kiểu nguyên
c = float(5)       # c là một biết kiểu thực
```

Nếu thực hiện lệnh print(a+b) thì Python sẽ báo lỗi vì không thể cộng một xâu với một số nguyên.

§6. PHÉP TOÁN, BIỂU THỨC, CÂU LỆNH GÁN

1. Phép toán

Các phép toán số học

Toán tử	Ý nghĩa	Ví dụ
+	Cộng	x + y+ 2
-	Trừ	x - y- 2
*	Nhân	x * y
/	Chia	x / y
%	Lấy phần dư của phép chia (mod)	x % y
//	Lấy phần nguyên của phép chia (div)	x // y
**	Lỹ thừa	x**y (x^y)

Các phép toán quan hệ

Toán tử	Ý nghĩa	Ví dụ
>	Lớn hơn	x > y
<	Nhỏ hơn	x < y
==	Bằng	x == y
!=	Khác	x != y
>=	Lớn hơn hoặc bằng	x >= y
<=	Nhỏ hơn hoặc bằng	x <= y

Các phép toán logic

Toán tử	Ý nghĩa	Ví dụ
and	Và: True khi cả hai đều True	x and y
or	Hoặc: True nếu một trong hai là True	x or y
not	Không: True khi False	not x

2. Biểu thức số học

Nội dung trình bày của phần này trong SGK Tin học 11 là phù hợp với Python. Riêng phép toán div, mod được thay thế bằng // và %.

3. Hàm số học chuẩn

Các hàm số học chuẩn được trình bày trong SGK được định nghĩa trong module **math**. Để sử dụng các hàm này trong Python chúng ta thực hiện lện import math.

Ví dụ. Trước khi dùng hàm sqrt() ta phải thực thiện import module math như sau:

```
import math
# với b = 4 thì lúc này giá trị được gán cho b là 2.
b = math.sqrt(a)
```

Lưu ý:

Hàm ln(x) được thay bằng math.log(x).

Hàm sqr(x) không có mà thay bằng x**2.

Hàm abs(x) có sẵn, không thuộc module math.

4. Biểu thức quan hệ

Nội dung trình bày của phần này trong SGK Tin học 11 là phù hợp với Python.

5. Biểu thức logic

Nội dung trình bày của phần này trong SGK Tin học 11 là phù hợp với Python.

6. Câu lệnh gán

Câu lệnh gán trong Python dùng dấu = để gán giá trị bên phải cho biến bên trái. Các lệnh gán trong Python cụ thể như sau:

Toán tử	Ví dụ	Tương đương với
=	x = 5	x = 5
+=	x += 5	x = x + 5
-=	x -= 5	x = x - 5
*=	x *= 5	x = x * 5
/=	x /= 5	x = x / 5
%=	x %= 5	x = x % 5
//=	x //= 5	x = x // 5
**=	x **= 5	x = x ** 5

§7. CÁC THỦ TỤC CHUẨN VÀO RA ĐƠN GIẢN

1. Nhập dữ liệu vào từ bàn phím

Trong Python để nhập liệu từ bàn phím ta dùng hàm input(). Giá trị nhập vào của hàm input() mặc định là kiểu xâu, do đó ta cần chuyển kiểu nếu như muốn lưu trữ giá trị nhập vào không phải kiểu chuỗi.

Ví dụ:

```
# nhập giá trị cho a (mặc định kiểu xâu)
a = input()
# nhập c (kiểu xâu)
b = input('Nhập b: ')
# nhập giá trị cho c có kiểu số nguyên.
c = int(input('Nhap c: '))
```

Nhập giá trị và nhấn Enter lần lượt cho từng biến.

2. Đưa dữ liệu ra màn hình

Python sử dụng hàm print() để hiển thị dữ diệu ra màn hình. Ta cũng có thể in thông báo nhập dữ liệu ra màn hình ngay trong lệnh input().

Ví dụ (tr.30 SGK)

Để nhập giá trị M từ bàn phím ta có thể code như sau:

```
print("Nhập giá trị M: ")
# sau khi nhập M sẽ được gán giá trị có kiểu str.
M = input()
```

Hoặc

```
N = int(input("Nhập số nguyên dương N <= 100: "))
```

Ví dụ (tr.31 SGK)

```
n = int(input("Lớp bạn có bao nhiêu người? "))
print("Vậy bạn có ", n-1, " người bạn trong lớp.")
input("Gõ Enter để kết thúc chương trình.")
```

Định dạng dữ liệu in ra màn hình

Ví dụ. Tính tổng 2 số nguyên nhập từ bàn phím

```
num1 = input('Nhập số thứ nhất: ')
num2 = input('Nhập số thứ hai: ')
# num1 và num2 hiện có kiểu str.
s = int(num1) + int(num2)
print('IN THÔNG BÁO KẾT QUẢ RA MÀN HÌNH:')
print('Tổng của hai số vừa nhập là:', s)
print('Cách 2: Tổng của {0} và {1} là {2}'.format(num1,
num2, s))
print('Cách 2-2: Tổng của {1} và {0} là
{2:.2f}'.format(num1, num2, s))

print(f"Cách 3: Tổng của {num1} và {num2} là {s}")
print(f"Cách 3-2: Tổng của {num1} và {num2} là {s:.2f}")
```

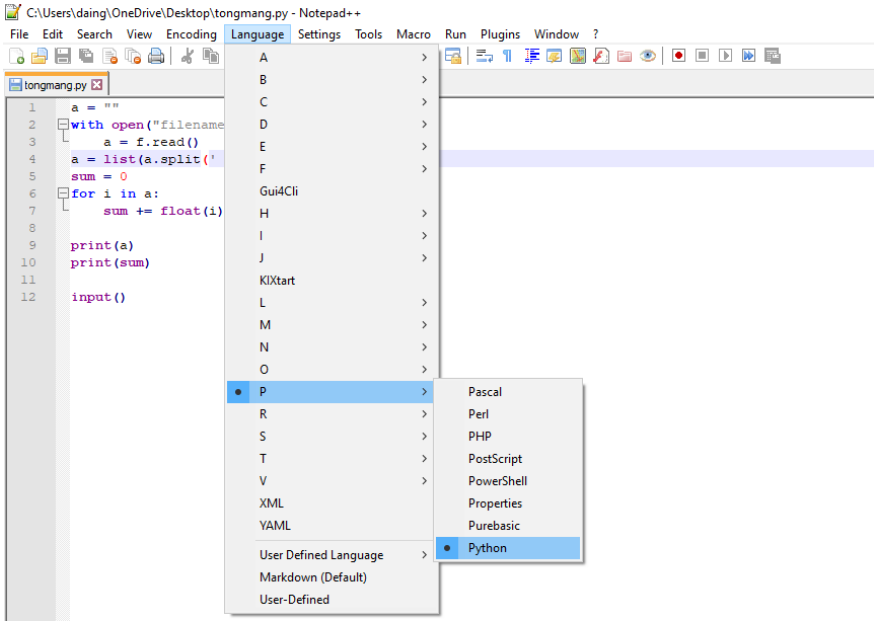
Các lệnh print() trong chương trình trên đều để hiển thị kết quả ra màn hình nhưng khác nhau về cách hiển thị. (hãy code và chạy thử chương trình để quan sát sự khác biệt giữa các cách in dữ liệu ra màn hình).

§8. SOẠN THẢO, DỊCH, THỰC HIỆN VÀ HIỆU CHỈNH CHƯƠNG TRÌNH

…

Với Python, ta có thể dùng các phần mềm IDE hoặc tex editor để soạn thảo và hiệu chỉnh chương trình.

Để code các chương trình đơn giản, không cần debug thì ta có thể dùng Notepad++ . (lưu ý chọn ngôn ngữ Python từ menu Language)

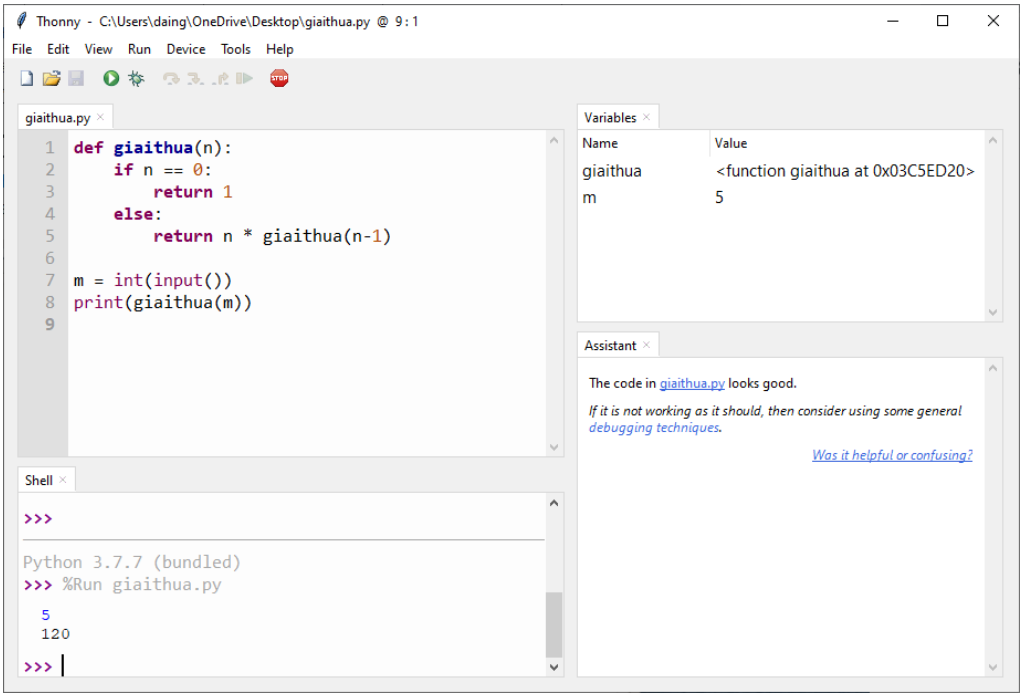


Để được hỗ trợ debug chương trình và các chức năng khác ta dùng các IDE để soạn thảo. Có nhiều IDE hỗ trợ lập trình Python như PyCham, Spyder, Thonny,… Trong chương trình tin học phổ thông chúng ta nên dùng Thonny làm công cụ soạn thảo lập trình Python vì một số yếu tố sau:

- Thonny có giao diện đơn giản, cấu hình nhẹ (trên cùng một máy khởi động nhanh hơn nhiều so với Pycham hay Spyder).

- Hỗ trợ debug trực quan giúp ta dễ theo dõi và hình dung quá trình thực thi chương trình.

- Sử dụng thư viện / module chuẩn của Python phát hành (không bổ sung hay import sẵn module).



Để thực thi chương trình (Run): Nhấn F5

Để Debug nhấn Ctrl+F5

Duyệt kiểm tra từng thao tác bằng F7

Xem thêm thông tin về thonny tại https://thonny.org/

BÀI TẬP VÀ THỰC HÀNH 1

1. Mục đích, yêu cầu

Sử dụng Thonny là trình soạn thảo để thực hành.

2. Nội dung

Thực hiện theo các yêu cầu SGK. Lưu ý các phím tắt để chạy chương trình Python trên Thonny là:

Chạy chương trình: F5

Kiểm thử - gỡ lỗi (debug): Ctrl+F5

Chương trình giải phương trình bậc 2.

```
from math import sqrt

print("Nhập a, b, c khác 0: ")
a = float(input())
b = float(input())
c = float(input())
d = b**2 - 4*a*c

if d < 0:
    print("Phương trình vô nghiệm.")
elif d == 0:
    print("Phương trình có 1 nghiệm: ", -b/2*a)
else:
    x1 = (-b - sqrt(d))/(2*a)
    x2 = -b/a - x1
    print("x1 = :", x1)
    print("x2 = :", x2)
input('Nhấn Enter để thoát!')
```

CÂU HỎI VÀ BÀI TẬP

Câu 4. Lưu ý trong Python biến sẽ tự thay đổi kiểu theo giá trị được gán vào.

Câu 5. Python không cấp phát bộ nhớ theo khai báo biến (không cần khai báo kiểu dữ liệu cho biến) mà sẽ tự động tùy biến theo dữ liệu được gán.

Câu 6. Lưu ý x^3 trong Python có thể viết là x**3

Câu 7. Hàm sqrt() thuộc module math. Khi sử dụng trong chương trình phải thực hiện câu lệnh: from math import sqrt.

Câu 9.

```
from math import pi
a = float(input("Nhập a: "))
s = (pi*a**2)/2
print(f"Diện tích nửa hình tròn là {s:.4f}")
```

Có thể thay pi bằng 3.14, tuy nhiên dùng pi từ module math sẽ cho kết quả chính xác hơn.

Câu 10.

```
from math import sqrt


h = float(input("Nhập độ cao h: "))
g = 9.8
v = sqrt(2*g*h)
print("Vận tốc khi chạm đất = ", v)
```

Chương III

CẤU TRÚC RẼ NHÁNH VÀ LẶP

§9. CẤU TRÚC RẼ NHANH

1. Rẽ nhánh

2. Câu lệnh if

```
if <điều kiện>: <câu lệnh>
```

Hoặc

```
if<điều kiện>:
      <câu hoặc khối lệnh>
```

Ví dụ:

```
if delta == 0:
      print("Phương trình vô nghiệm")
```

Câu lệnh if … else …

```
if<điều kiện>:
      <câu hoặc khối lệnh>
else:
      <câu hoặc khối lệnh>
```

Ví dụ:

```
a = 9
if (a % 3) == 0:
    print("a chia hết cho 3")
else: print("a không chia hết cho 3")
```

Câu lệnh if…elif…else …

```
if<điều kiện>:
      <câu hoặc khối lệnh>
elif<điều kiện n>:
      <câu hoặc khối lệnh>
else:
      <câu hoặc khối lệnh>
```

Ví dụ:

```
if d < 0:
    print("Phương trình vô nghiệm.")
elif d == 0:
    print("Phương trình có 1 nghiệm: ")
else:
    print("Phương tình có 2 nghiệm phân biệt")
```

3. Câu lệnh ghép

Câu lệnh ghép trong Python hay còn gọi là "khối lệnh" được thụt đầu dòng cùng một khoảng cách như nhau. (không cần dùng ngoặc hay từ khóa begin-end;).

Ví dụ:

```
if d < 0:
    print("Phương trình vô nghiệm.")
elif d == 0:
    print("Phương trình có 1 nghiệm: ")
else:
    print("Phương tình có 2 nghiệm phân biệt")
    print("X1 = ", x1)
    print("X2 = ", x2)
```

4. Một số ví dụ

Ví dụ 1.

```
from math import sqrt

print("Nhập a, b, c khác 0: ")
a = float(input())
b = float(input())
c = float(input())
d = b**2 - 4*a*c

if d < 0:
    print("Phương trình vô nghiệm.")
elif d == 0:
    print("Phương trình có 1 nghiệm: ", -b/2*a)
else:
```

```
    x1 = (-b - sqrt(d))/(2*a)
    x2 = -b/a - x1
    print("x1 = :", x1)
    print("x2 = :", x2)
```

Ví dụ 2.

```
n = int(input("Nhập năm: "))

if (n % 400 == 0) or ((n % 4 == 0) and (n%100 != 0)):
    print("Số ngày của năm", n, " là 366")
else:
    print("Số ngày của năm ", n, "là 365")
```

§10. CẤU TRÚC LẶP

1. Lặp

Nội dung trình bày của phần này trong SGK Tin học 11 là phù hợp với Python.

2. Lặp với số lần biết trước

Trước khi thực hiện ví dụ này cần cung cấp cho học sinh hàm range() để xác định một biến có các phần tử liên tục trong khoảng nào đó.

Range đầy đủ có 3 tham số range(start,end,step). Trong đó star = khởi đầu dãy, end = kết thúc dãy, step = bước nhảy (nếu bỏ trống thì mặc định bước nhảy là 1).

Ví dụ:

range(100) là dãy từ 0 đến 99.

range(2,10) là dãy liên tục từ 2 đến 9.

range(2,10,2) là dãy các số 2,4,6,8.

range(100,0,-1) là dãy liên tục từ 100 đến 1.

Ví dụ 1 (SGK,Tr.44)

```
# ví dụ Bài toán 1
a = int(input("Nhập a: "))
s = 0
for i in range(101):
    s += 1/(a + i)
print(s)
```

Vì sao lại là range(101)? Vì trong câu lệnh for i in range(101) thì i sẽ chạy từ 0 đến 100.

Ví dụ 2 (SGK,Tr.45)

```
# ví dụ Bài toán 1
m = int(input("Nhập M: "))
n = int(input("Nhập N: "))
mn = range(m,n+1)
s = 0
for i in mn:
    if ((i % 3 == 0)or(i % 5 == 0)):
        s += i
print(s)
```

3. Lặp với số lần chưa biết trước

Câu lệnh while

```
while <điều kiện >:
     <câu hoặc khối lệnh>
```

Lưu ý: Để thoát khỏi một vòng lặp vô hạn trong Python ta sử dụng tổ hợp phím Ctrl + C.

Ví dụ 1 (SGK,Tr.47)

```
a = int(input("Nhập a: "))
s = 1/a
n = 0
while not (1/(a+n) < 0.0001):
    n += 1
    s += 1/(a+n)
print("Tổng = ", s)
```

Ví dụ 2 (SGK,Tr.48)

```
m = int(input("Nhập m: "))
n = int(input("Nhập n: "))
while m != n:
    if m > n: m -= n
    else: n -= m
print("UCLN =", m)
```

BÀI TẬP VÀ THỰC HÀNH 2

Bài toán Số Pi-ta-go, Tr.49-50 SGK

```
a = int(input("Nhập a: "))
b = int(input("Nhập b: "))
c = int(input("Nhập c: "))
if (a**2 == b**2+c**2) or (b**2 == a**2+c**2) or (c**2 == a**2+b**2):
    print(a, b, c, "là bộ số pitago")
else:
    print(a, b, c, "không phải là bộ số pitago")
```

CÂU HỎI VÀ BÀI TẬP

Câu 1. So sánh 3 dạng câu lện if trong Python.

Câu 2. Nêu đặc điểm của khối lệnh trong Python?

Câu 3.

```
a = int(input("Nhập a: "))
s = 0
n = 1
while n <=100:
    s += 1/(a + n)
    n +=1
print(s)
```

Câu 4. a.

```
print("Nhập x, y:")
x = float(input())
y = float(input())

if (x**2 + y**2 <= 1):
    z = x**2 + y**2
elif (y >= x):
    z = x + y
else:
    z = 0.5
print(z)
```

Câu 4.b.

```
from math import sqrt
print("Nhập tọa độ x, y:")
x = float(input())
y = float(input())
print("Nhập tọa độ tâm O:")
a = float(input())
b = float(input())
r = float(input("Nhập bán kính r:"))
d = sqrt((x-a)**2 + (y-b)**2)
if r >= d:
    z = abs(x) + abs(y)
else:
    z = x +y
print("Z = ", z)
```

Câu 5. a

```
n = 1
Y = 0
while n <= 50:
    Y += n/(n+1)
    n += 1
print("Y = {:.3f}".format(Y))
```

Câu 5.b.

Nên chuyển sang sau bài hàm để xây dựng và gọi hàm tính giai thừa.

Câu 6.

```
# gọi
for x in range(37):
    for y in range(26):
        if (2*x + 4*y == 100) and (x+y == 36):
            print("Số gà - chó là:", x, y)
```

Câu 7.

```
# dùng while để kiểm tra dữ liệu nhập
cha = con = 0
while (cha - con < 25):
    cha = int(input("Nhập tuổi cha: "))
    con = int(input("Nhập tuổi con: "))
```

```
count = 0
while not con*2 == cha:
    con +=1
    cha +=1
    count +=1
print(count)
```

Câu 8.

```
a = float(input("Nhập số tiền gửi: "))
b = float(input("Số tiền muốn nhận: "))
lai = 0.003
tg = 1
tong = a+a*lai

while tong < b:
    tg += 1
    tong += a*lai

print(tg)
```

Chương 4.
KIỂU DỮ LIỆU CÓ CẤU TRÚC

§11. KIỂU MẢNG

Mảng là một cấu trúc dữ liệu cơ bản của tất cả các ngôn ngữ lập trình, nó là tập hợp các phần tử của một kiểu dữ liệu duy nhất, ví dụ mảng số nguyên, mảng xâu.

Python không có cấu trúc dữ liệu mảng mà sử dụng kiểu "Danh sách" (list) thay cho mảng. Không giống như mảng, mỗi list có thể lưu trữ phần tử với bất kỳ kiểu dữ liệu nào và làm được mọi thứ mà mảng có thể làm. Chúng ta có thể lưu trữ số nguyên, số thập phân, chuỗi trong cùng một list. Vì thế, làm việc với list khá linh hoạt.

List được biểu diễn bằng dãy các giá trị, được phân tách nhau bằng dấu phẩy, nằm trong dấu []. Các danh sách có thể chứa nhiều mục với kiểu khác nhau, nhưng thông thường là các mục có cùng kiểu.

Kiểu list có dạng:

```
a = [0,9,1,8,91,844,7]
```

Các phần tử trong list có chỉ số bắt đầu từ 0. Ta có thể gọi 1 hay nhiều phần tử trong list bằng các đối số trong ngoặc [].

Nâng cao: Ngoài việc xử lý mảng bằng list thuần túy ta có thể dùng module numpy. NumPy là thư viện nhằm phục vụ cho việc tính toán khoa học, hỗ trợ nhiều kiểu dữ liệu đa chiều giúp cho việc tính toán, lập trình, làm việc với các hệ cơ sở dữ liệu thuận tiện hơn.

1. Kiểu mảng một chiều

Trong Python, để xử lý dữ liệu mảng 1 chiều ta có thể dùng kiểu dữ liệu list. List trong Python không cần khai báo số phần tử, không cần khai báo kiểu dữ liệu. *(biến trong Python tự nhận biết kiểu dữ liệu được gán để tự động thay đổi kiểu cho phù hợp).*

Chương trình ví dụ trang 53 SGK:

```
print("Nhập vào nhiệt độ của 7 ngày: ")
t1 = float(input())
t2 = float(input())
t3 = float(input())
t4 = float(input())
t5 = float(input())
t6 = float(input())
t7 = float(input())
tb = (t1+t2+t3+t4+t5+t6+t7)/7
dem = 0
if t1 > tb: dem += 1
if t2 > tb: dem += 1
if t3 > tb: dem += 1
if t4 > tb: dem += 1
if t5 > tb: dem += 1
if t6 > tb: dem += 1
if t7 > tb: dem += 1
print(f"Nhiệt độ trung bình của tuần là: {tb:.2f}")
print("Số ngày có nhiệt độ cao hơn trung bình là: ", dem)
```

Chương trình 2

```
# ví dụ trang 54 SGK
n = int(input("Nhập số ngày: "))
a = []
for i in range(n):
    print("Nhập nhiệt độ ngày", i+1)
    ai = float(input())
    a += [ai]

tb = sum(a)/n
dem = 0
for j in a:
    if j > tb: dem +=1

print(f"Nhiệt độ trung bình của tuần là: {tb:.2f}")
print("Số ngày có nhiệt độ cao hơn trung bình là: ", dem)
```

Trong chương trình này ta đã sử dụng biết a có kiểu list để lưu nhiệt độ của các ngày (thay cho mảng một chiều truyền thống). Biến a không cần khai báo mà được khởi tạo bằng cách gán vào a một list rỗng [].

1.1 Khai báo

Như đã được học ở các bài trước. Biến trong Python không cần khai báo kiểu dữ liệu mà sẽ tự thay đổi kiểu cho phù hợp với giá trị được gán vào.

Một số lưu ý:

Khởi tạo một biến kiểu list chỉ bằng cách gán vào biến một list rỗng [] hoặc gán từ khóa list() cho biến.

Ví dụ:

```
ma = []
mb = list()
```

Để tạo một mảng (list) gồm n phần tử, các phần tử đều có giá trị rỗng thì ta có thể khai báo như sau:

```
ma = [None]*n
```

Chỉ số các phần tử trong danh sách (list) bắt đầu từ 0 và chiều ngược lại bắt đầu từ -1.

Có thể ghép các list với nhau bằng toán tử cộng +.

Cần lưu ý học sinh phân biện chỉ số phần tử và giá trị phần tử của list.

Ví dụ:

manga = ['P', 'Y', 'T', 'H', 'O', 'N']

Phần tử:	**P**	**Y**	**T**	**H**	**O**	**N**
Chỉ số	0	1	2	3	4	5
Chỉ số âm	-6	-5	-4	-3	-2	-1

1.2 Một số ví dụ

Ví dụ 1, trang 56 SGK

```
n = int(input("Nhập số lượng phần tử của dãy số, N = : "))
a = []
for i in range(n):
    print("Nhập phần tử thứ ", i+1)
    ai = int(input())
    a += [ai]

Max = a[0]
cs = 0
for j in a:
    if j > Max:
```

```
        Max = j
        cs_m = cs
    cs += 1

print("Giá trị lớn nhất của dãy là:", Max)
print("Chỉ số của Max là: ", cs_m)
```

Ví dụ 2

```
# ví dụ 2, trang 57 SGK
n = int(input("Nhập số lượng phần tử của dãy số, N = : "))
a = []
for i in range(n):
    print("Nhập phần tử thứ", i+1)
    ai = int(input())
    a += [ai]

i = n
while (i > 1):
    for j in range(n-1):
        if a[j] > a[j+1]:
           tam = a[j]
           a[j] = a[j+1]
           a[j+1] = tam
    i -= 1

print(a)
```

Trong Python có thể dùng hàm sort() để sắp xếp list mà không cần viết chương trình sắp xếp như trên. Viết lại chương trình trên như sau:

```
# ví dụ 2, trang 57 SGK (cách 2)
n = int(input("Nhập số lượng phần tử của dãy số, N = : "))
a = []
for i in range(n):
    print("Nhập phần tử thứ", i+1)
    ai = int(input())
    a += [ai]

a.sort()
print(a)
```

Ví dụ 3

```
# ví dụ 3, trang 59 SGK
n = int(input("Nhập số lượng phần tử của dãy số, N = : "))
a = []
for i in range(n):
    print("Nhập phần tử thứ", i+1)
    ai = int(input())
    a += [ai]
k = int(input("Nhập số k cần tìm : "))
# thêm dòng này để sắp xếp lại a nếu input chưa sắp xếp
# a.sort()
dau = 0
cuoi = n-1
tim = False
while (dau <= cuoi) and (not tim):
    giua = (dau+cuoi)//2
    if a[giua] == k:
        tim = True
    else:
```

```
        if a[giua] > k:
            cuoi = giua - 1
        else:
            dau = giua + 1

if tim:
    print("Chỉ số tìm được là ", giua)
else:
    print("Không tìm thấy")
```

Python cung cấp các hàm in, index để xử lý tìm kiếm trong list. Ta có thể áp dụng cho ví dụ trên như sau:

```
# ví dụ 3, trang 59 SGK (cách 2)
n = int(input("Nhập số lượng phần tử của dãy số, N = : "))
a = []
for i in range(n):
    print("Nhập phần tử thứ", i+1)
    ai = int(input())
    a += [ai]
k = int(input("Nhập số k cần tìm : "))

if k in a:
    cs = a.index(k)
    print(" Chỉ số tìm được là:", cs)
else:
    print("Không tìm thấy")
input()
```

2. Kiểu mảng hai chiều

Mảng 2 chiều có thể hiểu là mảng trong mảng. Giống như mảng một chiều mà trong đó mỗi phần tử của mảng cũng là một mảng một chiều.

Python cho phép tạo list trong list (Nested List). Vì vậy ta xử lý dữ liệu mảng 2 chiều bằng list như đã xử lý với mảng 1 chiều.

Ví dụ: manga là một mảng 2 chiều có 2 dòng, 3 cột như sau:

1	2	3
4	5	6

Ta tạo manga bằng nested list như sau:

```
manga = [[1,2,3],[4,5,6]]
```

Lưu ý, để thay thế mảng 2 chiều thì các phần tử của list con phải có cùng số phần tử.

Ví dụ 1. Tạo và in bảng nhân (SGK,Tr.61)

```
## tạo bảng 9 dòng x 10 cột
a = []
for i in range(1,10):
    a.append([])
    for j in range(1,11):
        x = i*j
        a[i-1].append(x)
## in
for i in range(9):
    for j in range(10):
        print(f'{a[i][j]:3}', end=' ')
    print()
```

Ví dụ 2 (SGK,Tr.62)

```
d=5; c=7
a = []
for i in range(d):
    a.append([])
    for j in range(c):
        x = int(input(f"Nhập phần tử a[{i}][{j}]: "))
        a[i].append(x)
k = int(input("Nhập k: "))
luu = []
for i in range(d):
    for j in range(c):
        if a[i][j] < k:
            luu.append(a[i][j])
if len(luu) > 0:
    print(f'DS các phần tử nhỏ hơn {k} là: ', end = '')
    for i in luu: print(i, end = ' ')
else: print(f'Không có phần tử nào nhỏ hơn {k}')
```

3. Các thao tác xử lý list

Phần nay được bổ sung để cung cấp thêm các thao tác xử lý với dữ liệu kiểu mảng (list).

a) Tạo list

```
# list rỗng
my_list = []

# list số nguyên
my_list = [1, 2, 3]

# list hỗn hợp
my_list = [1, "Hello", 3.4]

# list lồng hỗn hợp
my_list = ["mouse", [8, 4, 6], ['a']]
mang_1c = ['h', 'ọ', 'c', 'p', 'y', 't', 'h', 'o', 'n']
mang_2c = [[1,2,3],[4,5,6]]

print(mang_1c[2])          # Output: c
print(mang_1c[-2])         # Output: o

print(mang_1c[:3])         # Output: ['h', 'ọ', 'c']
print(mang_1c[:-6])        # Output: ['h', 'ọ', 'c']
print(mang_1c[5:])         # Output: ['t', 'h', 'o', 'n']
print(mang_1c[5:9])        # Output: ['t', 'h', 'o', 'n']

print(mang_2c[1][2])       # Output: 6
print(mang_2c[1][0])       # Output: 4
```

b) Thêm, sửa, xóa phần tử list

Thêm:

```
sole = [1, 3, 5]

sole.append(7)
# Output: [1, 3, 5, 7]
print(sole)
sole.extend([9, 11, 13])
# Output: [1, 3, 5, 7, 9, 11, 13]
print(sole)
sole = sole + [15,17]
# Output: [1, 3, 5, 7, 9, 11, 13, 15, 17]
print(sole)
sochan = [2, 4, 6]
sochan = sochan*3
# Output: [2, 4, 6, 2, 4, 6, 2, 4, 6]
print(sochan)
```

Sửa:

```
songuyen = [1,3,5,7,9,11]

songuyen[3] = 19
# Output: [1, 3, 5, 19, 9, 11]
print(songuyen)

songuyen[2:5] = [101,102,103]
# Output: [1, 3, 101, 102, 103, 11]
print(songuyen)
```

Xóa:

```
songuyen = [1,3,5,7,9,11]
del songuyen[3]
# Output: [1, 3, 5, 9, 11]
print(songuyen)

dayso = [1,3,5,7,9,11]
# xóa phần tử thứ 3,4
del dayso[3:5]
# Output: [1, 3, 5, 11]
print(dayso)
```

c) Các phương thức thường dùng với list

```
my_list = [3, 8, 1, 6, 0, 8, 4]

# Output: 3 (vị trí)
print(my_list.index(6))
# Output: 2 (có 2 phần tử = 8)
print(my_list.count(8))

# sắp xếp
my_list.sort()
# Output: [0, 1, 3, 4, 6, 8, 8]
print(my_list)

# đảo ngược list
my_list.reverse()
# Output: [8, 8, 6, 4, 3, 1, 0]
print(my_list)
```

```
# đảo ngược list
your_list = my_list[::-1]
# Output: [0, 1, 3, 4, 6, 8, 8]
print(your_list)

# xóa hết phần tử list
your_list.clear()
# Output: []
print(your_list)
```

BÀI TẬP VÀ THỰC HÀNH 3

Bài 1.

```
## Dùng module sinh số ngẫu nhiên
import random as rd
n = int(input("Nhập n: "))
a = []
for i in range(n):
    x = rd.randint(-300,300)
    a.append(x)
print(a)
k = int(input("Nhập số nguyên k: "))
s = 0
for i in a:
    if i%k == 0:
        s += i
print("Tổng cần tính là: ", s)
```

Bài 2.

```
n = int(input("Nhập n: "))
a = []
for i in range(n):
    x = int(input("Nhập phần tử a[{i}]: "))
    a.append(x)
print(a)
j = 1
for i in range(0,n):
    if a[i] > a[j]: j = i
```

```
print("Phần tử có giá trị lớn nhất là: ", a[j])
print("Chỉ số phần tử Max là: ", j)
```

Cách khác, sử dụng hàm có sẵn của Python để tìm max và chỉ số:

```
n = int(input("Nhập n: "))
a = [0]*n
for i in range(n):
    a[i] = int(input(f"Nhập phần tử a[{i}]: "))

print(a)
mx = max(a)
cs = a.index(mx)
print("Phần tử có giá trị lớn nhất là: ", mx)
print("Chỉ số phần tử Max là: ", cs)
```

BÀI TẬP VÀ THỰC HÀNH 4

Bài 1.

```
import random as rd
n = int(input("Nhập số lượng phần tử của dãy số, N = "))
a = [0]*n
for i in range(n):
    a[i] = rd.randint(-300,300)

print("Mảng vừa tạo:")
print(a)
i = n
while (i > 1):
    for j in range(n-1):
        if a[j] > a[j+1]:
           tam = a[j]
           a[j] = a[j+1]
           a[j+1] = tam
    i -= 1
print("Dã số đã được sắp xếp:")
print(a)
```

Bài 2.

```
import random as rd
n = int(input("Nhập số lượng phần tử của dãy số, N = "))
a = [0]*n
for i in range(n):
    a[i] = rd.randint(-300,300)
print("Mảng a:")
```

```
print(a)

b = []
for i in range(1,n+1):
    b.append(sum(a[:i]))

print("Mảng b: ")
print(b)
```

§12. KIỂU XÂU

Trong Python, xâu (string, từ khóa kiểu là str) là một dãy các ký tự Unicode. Unicode bao gồm mọi ký tự trong tất cả các ngôn ngữ và mang lại tính đồng nhất trong mã hóa.

Một xâu được đặt trong cặp nháy đơn hoặc nháy kép ', ". Xâu trên nhiều dòng được đặt trong cặp ba nháy đơn hoặc 3 nháy kép ''', """.

Một xâu trong Python được đánh chỉ số cho từng ký tự bắt đầu từ 0 và đảo chiều bằng cách thêm dấu âm - theo chiều ngược lại . Để truy cập vào phần tử xâu ta sử dụng tham số chỉ số trong ngặc vuông [] .

P	**Y**	**T**	**H**	**O**	**N**
0	1	2	3	4	5
-6	-5	-4	-3	-2	-1

1. Khai báo

Để tạo một biến có kiểu xâu ta chỉ cần gán xâu cho biến đó. Ví dụ:

```
d = '2021'
e = "2021"
f = ''' 2021 '''
```

Ép kiểu về xâu

```
# biến a có kiểu là int
a = 2021
# biến b, c đều có kiểu là str
b = str(a)
c = str(2021)
```

2. Các thao tác xử lý xâu

a) Ghép xâu (+)

Các xâu có thể ghép với nhau bằng toán tử cộng (+) hoặc nhân lên bằng toán tử nhân (*).

Ví dụ

```
str1 = 'Hà Nội'
str2 ='Việt Nam'

print(str1 + str2)
print(str1 * 3)
```

Kết quả:

```
Hà NộiViệt Nam
Hà NộiHà NộiHà Nội
```

b) So sánh xâu

Các nội dung trình bày trong SGK là phù hợp với Python.

Ví dụ

```
a = "Có phải em mùa thu Hà Nội"
b = "Có phải em là mùa thu Hà Nội"
print(a > b)       ## True
print(a < b)       ## False
```

Ký tự khác nhau đầu tiên của 2 xâu là chữ m trong "mùa" của xâu a và chữ l trong "là" của xâu b. ➔ *xâu a > b.*

c) Xóa ký tự trong xâu ?

Python **không** cho phép xóa ký tự trong xâu cho dù cho phép truy xuất ký tự theo chỉ số.

Ta có thể thay thế ký tự hoặc xâu con bằng hàm replace().

Ví dụ.

```
a = "Có phải em mùa thu Hà Nội"
b = a.replace('em ','')
print(a)
print(b)
```

Kết quả:

```
Có phải em mùa thu Hà Nội
Có phải mùa thu Hà Nội
```

d) Chèn ký tự vào xâu ?

Trong Python không thể chèn ký tự vào xâu thông qua chỉ số. Ta có thể chèn bằng cách thay thế với hàm replace() như mục trên.

e) Sao chép

Trong Python ta sử dụng thao tác cắt xâu để gán cho biến khác. Ví dụ:

```
a = "Có phải em mùa thu Hà Nội"
n = a[11:25]
print(n)
```

Kết quả in ra màn hình là: mùa thu Hà Nội

f) Độ dài xâu (len)

Để biết độ dài (số ký tự) của một xâu ta dùng hàm len()

Ví dụ:

```
a = "Có phải em mùa thu Hà Nội"
n = len(a)
```

```
print(n)
```

Kết quả: 25

g) Tìm kiếm trong xâu (find)

Python cung cấp hàng find() để thực hiện tìm kiếm và trả về vị trí (chỉ số) tìm thấy. Không tìm thấy sẽ trả về -1.

Ví dụ:

```
a = "Có phải em mùa thu Hà Nội"
n = 'Hà Nội'
k = a.find(n)
print(k)
```

Kết quả: 19

h) In hoa/ in thường (upper/lower)

```
a = "Có phải em mùa thu Hà Nội"
a = a.upper()
b = a.lower()
print(a)
print(b)
```

Kết quả:

```
CÓ PHẢI EM MÙA THU HÀ NỘI
có phải em mùa thu hà nội
```

i) Các thao tác thường dùng khác

1) Đưa giá trị biến khác vào trong xâu

```
a = 5
b = f"Biến a có giá trị là {a}"
print(b)
```

Kết quả:

```
Biến a có giá trị là 5
```

2) Tách/ ghép xâu thành list và ngược lại (split/join)

```
a = "Có phải em mùa thu Hà Nội"
b = a.split(' ')
c = ' '.join(b)
print(a)
print(b)
print(c)
```

Kết quả:

```
Có phải em mùa thu Hà Nội
['Có', 'phải', 'em', 'mùa', 'thu', 'Hà', 'Nội']
Có phải em mùa thu Hà Nội
```

3) Xử lý ký tự đặc biệt trong xâu

Các ký tự là dấu nháy trong xâu sẽ khiến chương trình hiểu là mở hoặc đóng một xâu. Ví dụ dấu nháy, dấu ngoặc kép trong câu: He said: “I’m studying Python”.

Ta có thể đặt câu này trong cặp ba nháy hoặc dùng \ để vô hiệu hóa dấu nháy như sau:

```
a = "He said \"I'm studying Python.\""
b = '''He said "I'm studying Python." '''
print(a)
print(b)
```

4) Kiểm tra sự tồn tại của ký tự/xâu con trong xâu

```
a = "Có phải em mùa thu Hà Nội"
b = "em"
print(b in a)
```

Kết quả: True

5) Vòng lặp trên xâu

Ví dụ, đếm số lần xuất hiện của chữ m trong câu “Có phải em là mùa thu hà nội”

```
a = "Có phải em là mùa thu Hà Nội"
count = 0
for i in a:
    if i == "m":
        count += 1
print(f"Chữ m xuất hiện {count} lần trong câu")
```

Kết quả:

```
Chữ m xuất hiện 2 lần trong câu
```

3. Một số ví dụ

Ví dụ 1

```
a = input("Nhập tên 1: ")
b = input("Nhập tên 2: ")
if len(a) > len(b):
    print(a)
else:
    print(b)
```

Ví dụ 2

```
a = input("Nhập xâu 1: ")
b = input("Nhập xâu 2: ")

if a[0] == b[len(b)-1]:
    print("Trùng nhau")
else:
    print("Khác nhau")
```

Ví dụ 3

```
a = input("Nhập xâu 1: ")
x = len(a)-1
for i in range(x,-1,-1):
    print(a[i], end='')
```

Ghi chú: Nếu quên có thể xem lại lệnh range() bài 10

Cách 2

```
a = input("Nhập xâu: ")
a = a[::-1]  ## đảo ngược xâu
print(a)
```

Ví dụ 4

```
a = input("Nhập xâu: ")
a = a.replace(' ','')
print(a)
```

Ví dụ 5.

```
s1 = input("Nhập xâu: ")
s2 = ''
for i in s1:
    if (i >= '0') and (i <= '9'):
        s2 +=i
print('Kết quả: ',s2)
```

BÀI TẬP VÀ THỰC HÀNH 5

Bài 1

```
a = input("Nhập xâu: ")
b = a[::-1]
if a == b:
    print("Xâu đối xứng")
else:
    print("Xâu không đối xứng")
```

Bài 2

```
a = input("Nhập xâu: ")
for i in a:
    count = 0
    for j in a:
        if i.lower() == j.lower(): count +=1
    print(f"Chữ {i} xuất hiện {count} lần")
```

Bài 2 (cách khác)

```
a = input("Nhập xâu: ")
at = a.lower()
b=''
for i in at:
    if i not in b:
        b += i
for i in b:
    print(f"Chữ {i} xuất hiện {at.count(i)} lần")
```

Bài 3

```
a = input("Nhập xâu: ")
if "anh" in a:
    b = a.replace("anh","em")
print(b)
```

§13. KIỂU BẢN GHI

Python không có kiểu bản ghi. Để xử lý dữ liệu kiểu bản ghi ta có thể kết hợp kiểu "từ điển" (dict) và list. Trong bài 11 (kiểu mảng) ta đã tìm hiểu về kiểu list trong Python. Ở bài này chúng ta tìm hiểu kiểu Từ điển (dict).

Dictionary là tập hợp các cặp khóa kèm giá trị không có thứ tự. Nó thường được sử dụng khi chúng ta có một số lượng lớn dữ liệu. Các dictionary được tối ưu hóa để trích xuất dữ liệu với điều kiện bạn phải biết được khóa để lấy giá trị.

1. Khai báo

Khởi tạo biến có kiểu từ điển rỗng (chưa gán giá trị)

```
d ={}
```

Khai báo dict có chỉ định key

```
diemso = {}.fromkeys(['toan', 'van', 'anh'], 0)
```

Lúc này, biến diemso sẽ có giá trị như sau:

```
{'toan': 0, 'van': 0, 'anh': 0}
```

2. Gán giá trị

Gán giá trị cho một biến từ điển (dict) được thực hiện theo cú pháp:

<tên biến>[<key>] = <Giá trị>

Ví dụ:

```
d = {}
d['ten'] = "Python"
d['phienban'] = 3.8
```

```
d['nam'] = 2020
print(d)
```

Output:

```
{'ten': 'Python', 'phienban': 3.8, 'nam': 2020}
```

Để xử lý ví dụ trong SGK trang 76,77. Ta dùng kết hợp list và dict như sau:

```
n = int(input('Nhập số học sinh: '))
ma=[]
for i in range(n):
    b = {}
    b['ten'] = input('Tên: ')
    b['ngs'] = input('Ngày sinh: ')
    b['dc'] = input('Địa chỉ: ')
    b['toan'] = int(input('Điểm Toán: '))
    b['van'] = int(input('Điểm Văn: '))
    t = b['toan']+b['van']
    if t >= 18:
        b['xl'] = 'A'
    elif t >= 14:
        b['xl'] = 'B'
    elif t >= 10:
        b['xl'] = 'C'
    else:
        b['xl'] = 'D'
    ma.append(b)
for i in range(n):
    print(ma[i]['ten'], '\t\t- Xếp loại: ', ma[i]['xl'])
```

3. Các thao tác cơ bản với kiểu dict

3.1 Truy xuất phần tử của dict

Ví dụ

```
bien_dict = {'ten': 'Nguyễn Lê Vi', 'tuoi': 2021}
print(bien_dict['ten'])
```

Kết quả:

```
Nguyễn Lê Vi
```

3.2 Thêm, sửa giá trị phần tử

Ví dụ

```
d = {'ten': 'Nguyễn Văn Văn', 'tuoi': 26}
print('Ban đầu:', d)
d['tuoi'] = 27
print('Sau khi sửa tuổi:', d)

d['diachi'] = 'Bình Phước'
print('Sau khi thêm địa chỉ: ', d)
```

Kết quả:

```
Ban đầu: {'ten': 'Nguyễn Văn Văn', 'tuoi': 26}
Sau khi sửa tuổi: {'ten': 'Nguyễn Văn Văn', 'tuoi': 27}
Sau khi thêm địa chỉ:  {'ten': 'Nguyễn Văn Văn', 'tuoi':
27, 'diachi': 'Bình Phước'}
```

3.3 Xóa phần tử

Chạy chương trình dưới đây và quan sát:

```
d = {1: 1, 2: 4, 3: 9, 4: 16, 5: 25}

# xóa và lấy giá trị phần tử được xóa
print(d.pop(2))
print(d)

# xóa phần tử cuối và trả về cặp key,value
print(d.popitem())
print(d)

# xóa tất cả các phần tử, trả về dict rỗng
d.clear()
print(d)
```

Kết quả:

```
4
{1: 1, 3: 9, 4: 16, 5: 25}
(5, 25)
{1: 1, 3: 9, 4: 16}
{}
```

Kiểu dữ liệu dict có thể sử dụng các hàm len(), sorted() như list. Ngoài ra dict còn có nhiều phương thức, phép toán khác – trong phạm vi thời lượng bài này không thể trình bày hết. Ta có thể tự tìm hiểu thêm.

CÂU HỎI VÀ BÀI TẬP

Có thể thay câu hỏi 1,2 bằng câu hỏi: Trong Python dùng kiểu dữ liệu nào để xử lý dữ liệu như kiểu mảng?

Câu 3. Các phần tử của list…

Câu 4. Tham chiếu đến phần tử list, phần tử của list lồng list (dạng thay thế mảng 2 chiều).

Câu 5.

```
n = int(input('Nhập n: '))
a = []
for i in range(n):
    nhap = True
    while nhap:
        t = int(input(f"Nhập giá trị a[{i}]: "))
        if abs(t) <= 1000:
            a.append(t)
            nhap = False
        else: nhap = True
print(a)
d = a[1]-a[0]

for i in range(n-1):
    if a[i+1] - a[i] != d:
        tb = 'Dãy không phải là cấp số cộng'
    else:
        tb = 'dãy là cấp số cộng'
print(tb)
```

Câu 6.

```
n = int(input("Nhập số phần tử: "))
ma =[]
for i in range(n):
    while True:
        t = int(input(f"Nhập phần tử ma[{i}]: "))
        if abs(t) <= 1000:
            ma.append(t)
            break

chan = le = nt = 0
snt = False
for i in ma:
    ## kiểm tra tính chẵn lẻ
    if i % 2 == 0:
        chan += 1
    else:
        le += 1
    ## kiểm tra số nguyên tố
    if i > 2:
        for j in range(2,int(i**0.5)+2):
            if i % j == 0:
                snt = False
                break
            else:
                snt = True
                print(i)
        if snt: nt +=1
print(f"Dãy số có {le} số lẻ, {chan} số chẵn và {nt} số
nguyên tố. ")
```

Bài 7.

```
n = int(input("Nhập số phần tử: "))
ma =[1,1]
for i in range(2,n):
    t = ma[i-1] + ma[i-2]
    ma.append(t)
#print(ma)
print(ma[n-1])
```

Cách khác (đệ quy):

```
n = int(input("Nhập số phần tử: "))
def F(n):
    if n == 1 or n == 2: return 1
    return F(n - 1) + F(n - 2)
print(F(n))
```

Bài 8.

```
import random as rd
n = int(input("Nhập số phần tử: "))
a=[]
for i in range(n):
    a.append([])
    for j in range(n):
        x = rd.randint(1,100)
        a[i].append(x)
for i in range(n):
    for j in range(n):
        print(f'{a[i][j]:3}', end=' ')
```

```
    print()
print('-'*20)
for i in range(n):
    for j in range(n):
        tem = a[i][j]
        a[i][j] = a[n-i-1][j]
        a[n-i-1][j] = tem

for i in range(n):
    for j in range(n):
        print(f'{a[i][j]:3}', end=' ')
    print()
```

Bài 9

```
import random as rd
n = int(input('Nhập n: '))
a = []
for i in range(n):
    a.append([])
    for j in range(n):
            t = rd.randint(-100,100)
            a[i].append(t)
for i in range(n):
    mx = max(a[i])
    for j in range(n):
        if a[i][j] == mx:
            a[i][j] = a[i][i]
            a[i][i] = mx
```

```
for i in range(n):
    for j in range(n):
        print(f'{a[i][j]:5}', end='')
    print()
```

Bài 10

```
s = input("Nhập xâu s: ")
count = 0
for i in s:
    if (i >= '0') and (i <= '9'):
        count += 1
print(f"Có {count} chữ số trong sâu s")
```

Bài 11. Đặt điều kiện khi print()

Chương V

TỆP VÀ THAO TÁC VỚI TỆP

§14. KIỂU DỮ LIỆU TỆP

1. Vai trò của kiểu tệp

2. Phân loại tệp và thao tác với tệp

Tệp (file) hay còn gọi là tập tin. File là tập hợp của các thông tin được đặt tên và lưu trữ trên bộ nhớ máy tính như đĩa cứng, đĩa mềm, CD, DVD,...

Ký tự lưu trong tệp không chỉ theo mã ASCII mà còn được ghi theo mã UNICODE.

Khi muốn đọc hoặc ghi file, chúng ta cần phải mở file trước. Khi hoàn thành, file cần phải được đóng lại để các tài nguyên được gắn với file được giải phóng.

Do đó, trong Python, một thao tác với file diễn ra theo thứ tự sau.

1. Mở tệp tin
2. Đọc hoặc ghi
3. Đóng tệp.

§15. THAO TÁC VỚI TỆP

1. Khai báo

Việc khai báo được thực hiện trong thao tác mở tệp để đọc/ghi dữ liệu.

2. Thao tác với tệp

Để thao tác với tệp (đọc hoặc ghi dữ liệu) trong Python có nhiều cách khác nhau. Trong đó sử dụng câu lện with là đơn giản và hiệu quả, ngắn gọn và an toàn. Cú pháp như sau:

```
with open("tenfile.txt","mode", endcoding) as tenbien:
   # Thực hiện các thao tác với file
```

Với câu lệnh trên file được mở để thao tác đồng thời sẽ tự động đóng file sau khi thao tác.

Các chế độ mở file (mode) thường dùng như sau:

Mode	Tác động
r	Mở chỉ để đọc. (mặc định)
w	Mở để ghi, nếu đã có nội dung thì sẽ ghi đè. Nếu file chưa tồn tại thì tạo file mới.
x	Tạo file độc quyền mới (exclusive creation) và ghi nội dung, nếu file đã tồn tại thì chương trình sẽ báo lỗi.
a	Mở file chế độ ghi tiếp. Nếu file đã tồn tại rồi thì nó sẽ ghi tiếp nội dung vào cuối file, nếu file không tồn tại thì tạo một file mới.
t	Mở file ở chế độ văn bản (mặc định)
b	Mở file ở chế độ nhị phân

2.1 Đọc file

Ví dụ, Ta có file ‘vao.txt’ được lưu cùng thư mục với file Python.

Mở file để đọc ta dùng phương thức **with open('tenfile', 'mode') as tenbien**. Ta có thể gán mode hoặc bỏ qua vì Python mặc định mở file là để đọc.

Đọc toàn bộ file

```
with open("vao.txt") as f:
   print(f.read())
```

Đọc từng dòng (1 dòng)

```
with open("vao.txt") as f:
    print(f.readline())
```

Đọc từng dòng (2 dòng)

```
with open("vao.txt") as f:
    print(f.readline())
    print(f.readline())
```

Đọc từng dòng đến hết file

```
with open("vao.txt") as f:
    for i in f:
        print(i, end='')
```

Lưu ý: Tất cả dữ liệu đọc từ file text đều có kiểu string. Vì vậy nếu là các chữ số, khi đọc vào phải ép kiểu thành number mới có thể thực hiện các phép toán số học.

2.2 Mở file để ghi

Mở file để ghi ta dùng phương thức **with open('tenfile', 'mode') as tenbien**. Mode thường dùng là **w** (ghi đè) hoặc **a** (ghi tiếp).

Cả mode **a** và **w** đều sẽ tạo file mới để ghi nếu file mở để ghi chưa tồn tại.

Ví dụ:

```
with open("ra.txt",'w') as f:
    f.write("Ghi noi dung nay vao file ")
    print("Ghi noi dung nay vao file cach 2")
```

Lưu ý: Để ghi dữ liệu vào file text ta chuyển về kiểu xâu trước khi ghi (nếu dữ liệu là số).

Chương trình sau sẽ báo lỗi:

```
a = "Ghi dong chu nay vao file ra.txt"
b = 994
with open("ra.txt",'w') as f:
    f.write(a + '\n')
    f.write(b)
```

Ta sửa lỗi như sau:

```
a = "Ghi dong chu nay vao file ra.txt"
b = 994
with open("ra.txt",'w') as f:
    f.write(a + '\n')
    f.write(str(b))
```

§16. VÍ DỤ LÀM VIỆC VỚI TỆP

Ví dụ 1

```
from math import sqrt

with open("TRAI.txt") as f:
    a = f.read()
a = a.split(' ')

_len = len(a)
tr = 0
for i in range(0,_len,2):
    d = sqrt(int(a[i])**2+int(a[i+1])**2)
    _d = round(d,2)
    tr += 1
    print(f"Khoảng cách đến các trại {tr} là: {_d:.2f}")
```

Ví dụ 2

```
with open("RESIST.DAT") as fi:
    tem = fi.read()
#print(tem)

# cắt xâu theo dòng để tạo list
a = tem.split('\n')
lena = len(a)
for i in range(lena):
    #cắt xâu theo khoảng trắng tạo list:
    a[i] = a[i].split(' ')
#print(a)
```

```
lenai = len(a[1])

with open('RESIST.EQU','w') as f2:
    for i in range(lena):
        for j in range(lenai):
            # chuyển str thành int để tính toán
            a[i][j] = int(a[i][j])
        r1 = a[i][0]
        r2 = a[i][1]
        r3 = a[i][2]
        r01 = r1*r2*r3/(r1*r2+r1*r3+r2*r3)
        r02 = r1*r2/(r1+r3) +r2
        r03 = r1*r3/(r1+r3) +r2
        r04 = r2*r3/(r2+r3) +r1
        r05 = r1+r2+r3
# ghi vào file
        f2.write(str(r01)+' ')
        f2.write(str(r02)+' ')
        f2.write(str(r03)+' ')
        f2.write(str(r04)+' ')
        f2.write(str(r05)+'\n')
print('Hoàn thành ghi dữ liệu vào file!')
```

Chương VI

CHƯƠNG TRÌNH CON VÀ LẬP TRÌNH CÓ CẤU TRÚC

§17. CHƯƠNG TRÌNH CON VÀ PHÂN LOẠI

1. Khái niệm chương trình con

Chương trình con trong Python gồm các packpage, module và các hàm được xây dựng sẵn (build-in) hoặc do người lập trình tự xây dựng. Packpage là tập hợp nhiều module, một module là tập hợp nhiều hàm. *(một packpage có thể chỉ có 1 module, một module có thể chỉ có 1 hàm – nhưng không ai làm thế)*

Module trong Python tương tự như thư viện trong Pascal. Module là một file chứa các hàm liên quan. Ví dụ, module math là một file có tên math.py chứa các hàm toán học chuẩn; module random là file random.py chứa các hàm sinh dữ liệu ngẫu nhiên.

Chương trình ví dụ trang 92 SGK.

Thay vì nhập các số từ bàn phím, ta điều chỉnh yêu cầu đề bài để ứng dụng kỹ thuật thao tác với file, string, list như sau:

Cho file input.txt có 2 dòng, dòng 1 gồm các biến: a,b,c,d; dòng 2 gồm các biến: m,n,p,q. Các biến cách nhau bởi một dấu phẩy.

```
def luythua(a,b):
    return a**b

with open('input.txt') as f:
    l1 = f.readline().replace('\n','')
    l2 = f.readline().replace('\n','')
a = l1.split(',')
b = l2.split(',')

tong = 0
for i in range(len(a)):
    a[i] = float(a[i])
    b[i] = int(b[i])
    tong += luythua(a[i],b[i])
print(tong)
```

2. Phân loại và cấu trúc của chương trình con

2.1 Phân loại

Python không phân biệt thủ tục với hàm.

Trong Python có thể phân làm 2 loại chương trình con. (1) là các chương trình con xây dựng sẵn (build-in) và (2) chương trình con do người lập trình tự xây dựng.

Về cấu trúc, hàm trong Python cũng chia làm 2 loại là (1) hàm thông thường, được đặt tên và (2) "hàm ẩn danh", không có tên (hàm lambda).

Về quy mô ta có thể phân thành 3 loại: hàm, module và packpage.

2.2 Cấu trúc

Cấu trúc hàm trong Python:

```
def <tenham>(<biến/danh sách biến>):
   """docstring (ghi chú về chức năng của hàm/
   hàm không nhất thiết phải có docstring)"""
   <lệnh/ khối lệnh thực hiện nhiệm vụ>
```

Hàm ẩn danh

```
<tên biến> = lambda < tham biến>: <biểu thức, giá trị trả
về>
```

Ví dụ. Hàm tính tổng 3 số

```
def tong1(x,y,z):
    t = x + y + z
    return t

def tong2(x,y,z):
    return x + y + z

tong3 = lambda x,y,z: x + y + z

a = 5
b = 6
c = 7

print(tong1(a,b,c))
print(tong2(a,b,c))
print(tong3(a,b,c))
```

Kết quả: cả 3 lệnh print đều có kết quả là 18

2.3 Thực hiện chương trình con

Đối với hàm:

```
<tên hàm>(<tham số>)
```

Những hàm không có tham số thì không cần truyền tham số.

Ví dụ:

```
sum(a,b)   # tính tổng a, b
abs(a)     # lấy giá trị tuyệt đối của a
tenham()   # gọi một hàm không truyền tham số
```

Đối với module

```
import <tên module>
# hoặc
from <tên module> import <tên hàm>
# hoặc
from <tên module> import *         # import tất cả các hàm
```

Ví dụ, để sử dụng hàm khai căn bậc hai của một số ta phải dùng hàm sqrt() từ module math, ta khai báo trước khi sử dụng như sau:

```
import math
form math import *
# Hai câu lệnh trên là tương đương
# chỉ import một hàm từ module
from math import sqrt
```

Đối với Packpage (đọc thêm)

Ví dụ tạo và sử dụng package với các module/hàm trong module:

1. Tạo thư mục

```
import os
```

```
os.mkdir("dnx")
```

2. Đặt các module khác nhau trong package:

module1.py

```
def inra():
    print("Được in từ hàm inra trong module 1")
```

module9.py

```
def msg():
    print("Dòng này in từ hàm msg của module 9")
def msg2():
   print("Dòng này in từ hàm msg2 của module 9")
```

3. Tạo một __init__.py file.

```
# file __init__ trống
import dnx.module1
import dnx.module9
```

4. Import package này và gọi hàm từ các module.

```
from dnx import *
module1.inra()
module9.msg2()
module9.msg()
```

Kết quả là:

```
Được in từ hàm inra trong module 1
Dòng này in từ hàm msg2 của module 9
Dòng này in từ hàm msg của module 9
```

Lưu ý thêm: Ta cũng có thể viết hàm trực tiếp trong file __init__.py.

§18. VÍ DỤ CÁCH VIẾT VÀ SỬ DỤNG CHƯƠNG TRÌNH CON

Chương trình vẽ hình chữ nhật (SGK,Tr.96)

```
def vehcn():
    print('*'*50)
    for i in range(10):
        print('*' + ' '*48 + '*')
    print('*'*50)
# gọi hàm
vehcn()
```

Chương trình vẽ hình chữ nhật có tham số (SGK.Tr.98)

```
def vehcn(a,b):
    print('*'*a)
    for i in range(b):
        print('*' + ' '*(a-2) + '*')
    print('*'*a)

a = int(input('Nhập chiều dài: '))
b = int(input('Nhập chiều rộng: '))
vehcn(a,b)
```

Chương trình hoán đổi 2 biến (SGK. Tr.100)

```
a = int(input('Nhập a: '))
b = int(input('Nhập b: '))
print('Trước khi đổi:', a, b)
```

```
a,b = b,a
print('Sau khi đổi  :', a, b)
```

Ví dụ 1 (SGK, Tr.101)

```
def ucln(a,b):
    while b != 0:
        t = a % b
        a = b
        b = t
    return a

a = int(input('Nhập tử : '))
b = int(input('Nhập mẫu: '))
x = ucln(a,b)
a = a // x
b = b // x
print(f'Phân số đã rút gọn là {a}/{b}')
```

Ví dụ 2 (SGK, Tr.102)

```
# Python cung cấp sẵn hàm min()

def nhonhat(a,b):
    if a < b: return a
    else: return b

a = float(input('Nhập a: '))
b = float(input('Nhập b: '))
c = float(input('Nhập c: '))
```

```
m = nhonhat(nhonhat(a,b),c)
print(f'Giá trị nhỏ nhất của {a}, {b}, {c} là {m}')
```

BÀI TẬP VÀ THỰC HÀNH 6

1. Mục đích, yêu cầu

Bỏ yêu cầu chữ chạy trên màn hình.

2. Nội dung

Python cung cấp sẵn phươn thức đảo ngược xâu, list. Ví dụ:

```
a = "HỌC LẬP TRÌNH PYTHON"
b = a[::-1]
print(a)
print(b)
```

Kết quả:

```
HỌC LẬP TRÌNH PYTHON
NOHTYP HNÌRT PẬL CỌH
```

Viết và chạy thử các ví dụ thao tác xử lý xâu trong phần §12.2

BÀI TẬP VÀ THỰC HÀNH 7

1. Mục đích, yêu cầu

2. Nội dung

Bài thực hành này tập trung thực hành sử dụng kiểu dữ liệu record. Tuy nhiên, như đã giới thiệu ở bài §13, kiểu bản ghi hoàn toàn có thể xử lý bằng list, dict. Vì vậy trong bài này chúng ta sử dụng list để giải quyết các bài toán đã nêu.

Chương trình SGK, Tr.106

```
from math import sqrt
def nhapdiem():
    ma = []
    for i in range(3):
        ma.append([])
        x = int(input('Nhập x: '))
        y = int(input('Nhập y: '))
        ma[i].append(x)
        ma[i].append(y)
    return ma
def kh_cach(a,b):
    return sqrt((a[0]-b[0])**2 + (a[1]-b[1])**2)
def daicanh(x):
    a = kh_cach(x[0],x[1])
    b = kh_cach(x[1],x[2])
    c = kh_cach(x[0],x[2])
    return a, b, c
def chuvi(x):
    cv = 0
```

```
    for i in daicanh(x):
        cv += i
    return cv
def dientich(x):
    p = chuvi(x)/2
    dc = daicanh(x)
    s = sqrt(p*(p-dc[0])*(p-dc[1])*(p-dc[2]))
    return s
def tinhchat(x):
    tc = ''
    d = daicanh(x)
    if (d[0] == d[1]) and (d[1] == d[2]):
        tc = 'Tam giác đều'
    elif (d[0] == d[1]) or (d[1] == d[2]):
        tc = 'cân'
    elif (d[0]**2+ d[1]**2 == d[2]**2) or (d[0]**2+
d[2]**2 == d[1]**2) or (d[1]**2+ d[2]**2 == d[0]**2):
        tc = 'Tam giác vuông'
    else:
        tc = 'Tam giác thường'
    return tc

tg = nhapdiem()

print('Tọa độ 3 đỉnh của tam giác là: ', tg)
print('Tính chất:', tinhchat(tg))
print(f'Chu vi tam giác = {chuvi(tg):9.3f}')
print(f'Diện tích của tam giác = {dientich(tg):9.3f}')
```

Câu c, SGK, Tr.108

```
from math import sqrt
with open('in.txt') as f:
    n = int(f.readline())
    a = []
    for i in range(n):
         at = list(map(int,f.readline().split()))
         a.append(at)

def kh_cach(xa,ya,xb,yb):
    return sqrt((xa-xb)**2 + (ya-yb)**2)
def daicanh(x):
    a = kh_cach(x[0],x[2],x[1],x[3])
    b = kh_cach(x[2],x[4],x[3],x[5])
    c = kh_cach(x[0],x[2],x[3],x[5])
    return a, b, c
def chuvi(x):
    cv = 0
    for i in daicanh(x):
        cv += i
    return cv
def dientich(x):
    p = chuvi(x)/2
    dc = daicanh(x)
    s = sqrt(p*(p-dc[0])*(p-dc[1])*(p-dc[2]))
    return s
def tinhchat(x):
    tc = ''
    d = daicanh(x)
    if (d[0] == d[1]) and (d[1] == d[2]):
        tc = 'đều'
    elif (d[0] == d[1]) or (d[1] == d[2]):
```

```
        tc = 'cân'
    elif (d[0]**2+ d[1]**2 == d[2]**2) or (d[0]**2+
d[2]**2 == d[1]**2) or (d[1]**2+ d[2]**2 == d[0]**2):
        tc = 'vuông'
    else:
        tc = 'thường'
    return tc

deu = can = vuong = thuong = 0
for i in range(n):
    tg = a[i]
    t = tinhchat(tg)
    if t == 'đều': deu += 1
    elif t == 'cân': can += 1
    elif t == 'vuông': vuong += 1
    else: thuong += 1

with open('TAMGIAC.OUT','w') as fo:
    fo.writelines(str(deu))
    fo.write('\n')
    fo.write(str(can))
    fo.write('\n')
    fo.write(str(vuong))
    fo.write('\n')
    fo.write(str(thuong))
```

§19. THƯ VIỆN CHƯƠNG TRÌNH CON CHUẨN

1. Module

Ngoài việc xây dựng chương trình con (hàm) trực tiếp trong chương trình. Python cung cấp hệ thống chương trình con dưới dạng module. Mỗi Module là một file, trong đó các lớp, hàm và biến được định nghĩa. Tất nhiên, một Module cũng có thể bao gồm code có thể chạy.

Chương trình con chuẩn có thể hiểu là các Module đã được xây dựng sẵn trong Python. Đó là math, random, threading, collections, os, mailbox, string, time, … Mỗi Module này đã định nghĩa sẵn rất nhiều hàm để bạn có thể sử dụng để thực hiện các tính năng khác nhau.

Ví dụ sử dụng math Module:

```
import math
a=4.6
print (math.ceil(a))
print (math.floor(a))
b=9
print (math.sqrt(b))
print (math.exp(3.0))
print (math.log(2.0))
print (math.pow(2.0,3.0))
print (math.sin(0))
print (math.cos(0))
print (math.tan(45))
```

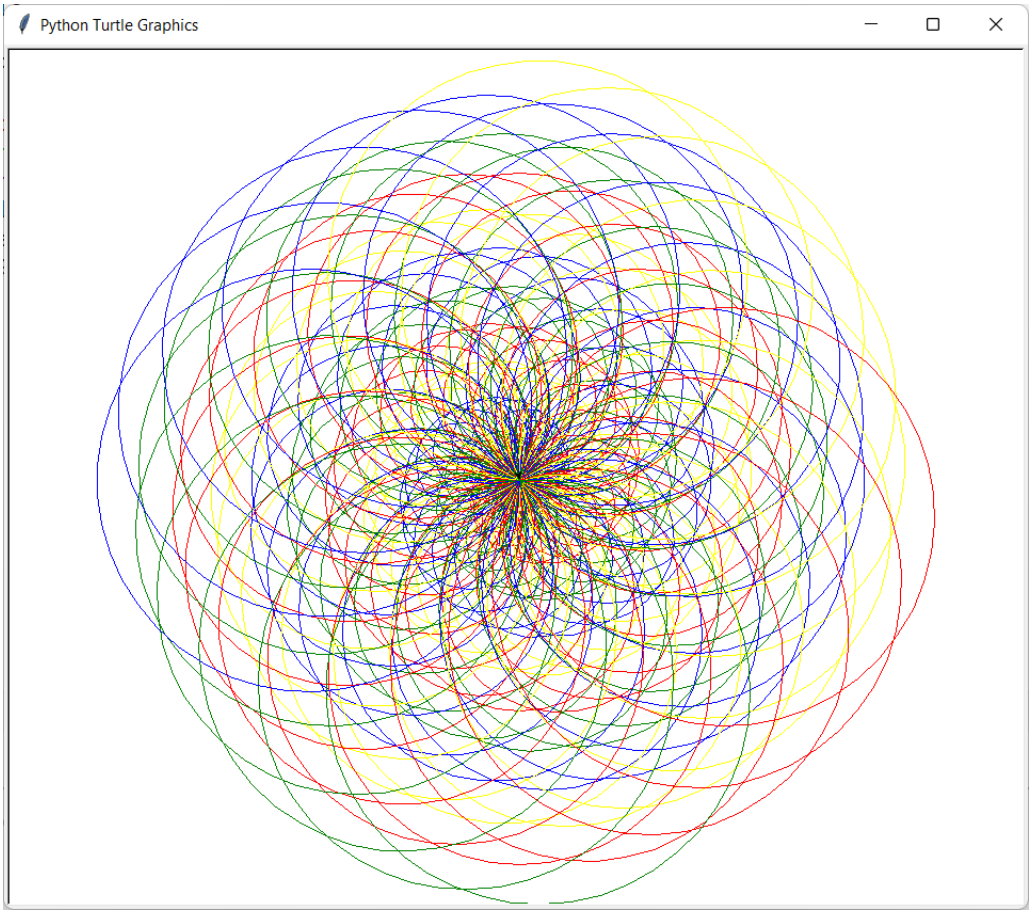
Ví dụ sử dụng random Module:

```
import random
print (random.random())
print (random.randint(2,8))
```

2. Package

Một Package là một tập hợp các Module, sub-package, … tương tự nhau. Đó là một cấu trúc có thứ bậc của thư mục và file.

3. Thư viện đồ họa Turtle



Hình vẻ bởi 200 vòng tròn

Turtle Graphics là một chương trình con có sẵn trong Python hỗ trợ hiển thị giao diện đồ họa. Cùng khảo sát một số ví dụ sau đây.

```
#vẽ hình vuông
import turtle
t = turtle.Pen()
```

```
for x in range(100) :
    t.forward(x)
    t.left(90)
```

```
# vẽ hình xoắn ốc cuốn
import turtle
t = turtle.Pen()
for x in range(100) :
    t.forward(x)
    t.left(91)
```

```
# hình xoắn ốc cuốn tròn
import turtle
t = turtle.Pen()
for x in range(100) :
    t.circle(x)
    t.left(91)
```

```
#vẽ xoắn ốc tròn 4 màu nền đen
import turtle
t = turtle.Pen()
turtle.bgcolor("black")
colors = ["red", "yellow", "blue", "green"]
for x in range(100):
    t.pencolor(colors[x%4])
    t.circle(x)
    t.left(91)
```

```
import turtle
t = turtle.Pen()
t.speed(0)
turtle.bgcolor("black")
colors = ["red", "yellow", "blue", "green"]
for x in range(100):
    t.pencolor(colors[x%4])
    t.circle(x)
    t.left(45)
```

BÀI TẬP VÀ THỰC HÀNH 7

1. Mục đích, yêu cầu

Như SGK

2. Nội dung

2.1 Câu a

```
# đường gấp khúc ngẫu nhiên
import random as rd
import turtle as t
t.bgcolor('black')
color = ['red','white','yellow','blue']
for i in range(100):
    t.speed(0)
    t.fd(50)
    n = rd.randint(i,360)
    if n%2 == 0:
        t.left(n)
    else:
        t.right(n)
    t.pencolor(color[i%4])
```

2.2 Câu b

```
import turtle as t
t.bgcolor('black')

def vuong(d):
    t.pencolor('white')
    t.pendown()
    for i in range(4):
```

```
        t.fd(d)
        t.left(90)
    t.penup()

def chunhat(d,r):
    t.pendown()
    t.pencolor('yellow')
    for i in range(2):
        t.fd(d)
        t.left(90)
        t.fd(r)
        t.left(90)
    t.penup()

def tron(r):
    t.pendown()
    t.pencolor('green')
    t.circle(r)
    t.penup()

def elip(r):
    t.pendown()
    t.pencolor('red')
    for i in range(2):
        t.circle(r,90)
        t.circle(r//2,90)
    t.seth(-45)
    t.penup()

t.penup()
t.goto(250,200)
tron(100)
```

```
t.goto(-250,200)
vuong(200)
t.goto(150,-150)
chunhat(200,150)
t.goto(-200,-150)
elip(120)
```

CÂU HỎI VÀ BÀI TẬP

Câu 1. Python không phân biệt

Câu 2. Hàm in ra màn hình một xâu

Câu 3. Hàm trả về nhiều giá trị

Câu 4.

```
from math import sqrt

def ucln(a,b):
    while b > 0:
        t = a % b
        a = b
        b = t
    return a
def bcnn(a,b):
    t = a*b // ucln(a,b)
    return t

a = int(input('Nhập a: '))
b = int(input('Nhập b: '))
print(f'Ước chung lớn nhất của {a} và {b} là {ucln(a,b)}')
print(f'Bội chung nhỏ nhất của {a} và {b} là {bcnn(a,b)}')
```

Cải tiến: Viết thêm và gọi hàm nhập a, b cho chương trình.

Tìm hiểu thêm về Python

Nội dung trình bày trong tài liệu này chỉ là phần bám sát Sách giáo khoa Tin học 11 nhằm phục vụ thầy cô giáo và các bạn học sinh lớp 11 chuyển đổi ngôn ngữ lập trình, tiếp cận chương trình Giáo dục phổ thông 2018. Vì vậy, để có hiểu thấu đáo và sử dụng Python trong lập trình thì chúng ta cần tiếp tục tìm hiểu các tài liệu khác.

Tài liệu tiếng Anh

https://www.python.org/

https://www.w3schools.com/python/default.asp

https://www.programiz.com/python-programming

Các tài liệu cùng tác giả có trên website: www.dainganxanh.com

https://python.dainganxanh.com/

Liên hệ tác giả: dainganxanh

Email: dainganxanh@msn.com